Vol.2015-CG-158 No.1
2015/2/27

SPH法と Shallow Water モデルによる高速な流体シミュレーション

仲田 拓也[1]　藤澤 誠[1,a)]　三河 正彦[1,b)]

概要：本論文では大規模なシーンを高速に計算をするための流体シミュレーション手法を提案する．提案手法では，2 次元の SPH (Smoothed Particle Hydrodynamics) 法を用いた Shallow Water シミュレーションと 3 次元の SPH 法を組み合わせることで，高速かつ 3 次元的な動きにも対応する．2 次元パーティクルから 3 次元パーティクルを生成し，運動量を保存するように相互作用を計算することで現実的な振る舞いを実現している．また，提案手法を様々なシーンで実験することでその有効性を確認した．

1. はじめに

近年，自然現象をコンピューターグラフィックスで再現するために物理シミュレーションが用いられている．その中でも津波や洪水，爆発などの流体現象を再現するために流体シミュレーションが使われている．ゲームなどのインタラクティブなアプリケーションでは，高精細かつリアルタイムな計算速度が求められているが，高精細な流体シミュレーションは計算コストが高く，特に川や湖や海などの大規模な流体シミュレーションをリアルタイムで計算することは難しい．このような流体シミュレーションを実現するための方法の一つに Shallow Water モデルによる流体シミュレーションがある．この方法は 3 次元流体の運動計算を 2 次元の流体表面の運動計算だけで近似することで計算コストを減らした方法であり，高速に計算が可能である．しかし，流体表面の上下運動しか扱えないため，しぶきや大きな波が作り出すトンネル形状など，水が覆いかぶさるような 3 次元的な挙動を表現できないということが問題となる．

本論文では 2 次元のパーティクル法を用いた Shallow Water モデルによる波のシミュレーションに対して，3 次元的な動きをパーティクル法の一種である SPH(Smoothed Particle Hydrodynamics) 法により追加することで，より高精細で高速な流体シミュレーションを実現する方法を提案する．提案手法では Shallow Water モデルにより，大規模なシーンでも高速に計算可能であり，かつ，従来手法では実現できなかった 3 次元的な挙動を 3 次元の SPH 法と組み合わせることで実現する．

[1] 筑波大学
University of Tsukuba
[a)] fujis@slis.tsukuba.ac.jp
[b)] mikawa@slis.tsukuba.ac.jp

2. 関連研究

2.1 流体シミュレーションに関する関連研究

流体シミュレーションを行うために，オイラー的手法の一つであるグリッド法 [1] そして，ラグランジュ的手法の一つであるパーティクル法についての研究が盛んに行われている．パーティクル法の中でも計算速度，実装の容易さから SPH 法 [2] が一般的に用いられている．SPH 法を用いた流体シミュレーションにおいて，流体の質量保存はパーティクル数が変化しない限り満たされる．しかしパーティクル同士が重なり合うことで，流体が圧縮してしまうという問題点がある．この問題を解決するために WCSPH 法 [3] や PCISPH 法 [4] といった非圧縮性を保つ手法が提案されている．Macklin ら [5] は初期密度を保つような拘束条件から，パーティクルの位置を直接修正することにより高い非圧縮性を保つ手法を提案した．この手法は非圧縮性を保ちつつインタラクティブな速度を実現可能であるため，本論文では 3D パーティクルの運動計算にこの手法を用いる．

2.2 リアルタイム流体シミュレーションに関する関連研究

リアルタイム流体シミュレーションを実現するために，さまざまな手法が提案されている．計算コストを減らすために，3 次元の流体シミュレーションの代わりに，水面を表す 2 次元高さ場を用いる方法がある．Kass ら [6] は波動方程式により高さ場を動かすことでリアルタイムシミュレーションを実現した．しかしこの手法は運動量が考慮さ

れていないため，物理的な表現とはなっていない．Layton ら [7] はナビエ・ストークス方程式に基づく速度場を用いて，2 次元のグリッド法により Shallow Water Equation(以下 SWE) を解く方法を提案した．Lee ら [8] と Solenthaler ら [9] はパーティクルを用いて SWE を解くシミュレーション手法を提案した．しかし，Shallow Water モデルは水面のシミュレーションであるため，流体が覆い被さるような 3 次元的な挙動を表現することができない．Chentanez ら [10] はグリッド法による Shallow Water モデルに対して，しぶきなどの 3 次元的な挙動がおこる箇所を検出し，その箇所にパーティクルを生成することにより，高精細なリアルタイム流体シミュレーションを実現した．また Thurey ら [11] は波が覆いかぶさる部分を検出し，その箇所にメッシュを生成，そのメッシュを移流させることで波が覆い被さる 3 次元的な挙動を実現した．Chentanez ら [12] は Shallow Water モデルに 3 次元のグリッド法を組み合わせることにより高精細なしぶき表現を実現した．Wang ら [13] は Shallow Water モデルに表面張力を追加する手法を提案している．これらの方法はいずれも限定されたシーンでのみ効果を発揮するものである．

本論文では 2D パーティクルによる Shallow Water シミュレーションに対して，波の覆い被さり，しぶきなどの 3 次元的な挙動を，3D パーティクルを追加することにより実現する方法を提案する．水面と 3 次元的な挙動の両方がにパーティクルで表現されるため，運動計算やレンダリングの際に親和性が高く，違和感のないシミュレーションが可能となる．また 3D パーティクルの運動量を保存するように，2D パーティクルとの相互作用を計算することで現実的な挙動を実現している．

3. 提案手法

本論文では Shallow Water モデルを用いて 2 次元のパーティクル法により波のシミュレーションを行う (§3.1)．この 2D シミュレーションでは，砕け波やしぶきなど 3 次元的な動きは再現できない．そこで 2D シミュレーション結果をもとに 3D パーティクルの生成 (§3.3) および削除 (§3.4) を行い，その運動計算を行う (§3.2) ことでこれらの現象を追加する．最後に 2D パーティクルと 3D パーティクルをあわせてレンダリングする (§3.5)．これらを各タイムステップ毎に実行する．以降の節で処理の詳細を説明していく．

3.1 2D パーティクルシミュレーション

しぶきなどの 3 次元的な挙動を 3D パーティクルで表現するため，グリッド法ではなくパーティクル法による Shallow Water シミュレーションを用いる．本論文では 2D パーティクルにより SWE を解くことで全体的な動きを求める [9]．重力は y 軸に対して加わると仮定し，2D パーティクルは xz 平面上に配置され，xz 平面内での 2 次元的な動きから高さを求める．

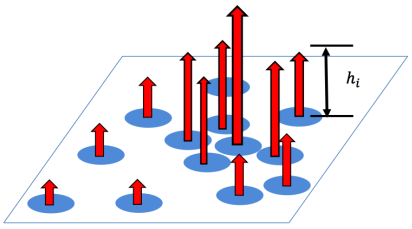


図 1 パーティクルに対する高さ定義

3.1.1 2D パーティクルによる Shallow Water シミュレーション

Shallow Water モデルによるシミュレーションでは，シミュレーション空間をグリッド分割し，グリッドごとに定義された高さ場を，速度場に従って変化させることでシミュレーションを行う [7]．SWE はナビエ・ストークス方程式に基づいた式であり，水面の運動を表している式である．SWE を解くことで，流体の速度場を更新し，その速度をもとに高さ場を変化させ，シミュレーションを行う．SWE は流体の質量と運動量の保存を表しており以下のように記述できる．

$$\frac{\partial h}{\partial t} = -\nabla \cdot (h\mathbf{u}) \tag{1}$$

$$\frac{D\mathbf{u}}{Dt} = -g\nabla(h + H(\mathbf{x})) + a_{ext} \tag{2}$$

ここで h は水深，$\mathbf{u}$ は流体の速度，g は重力，$H(\mathbf{x})$ は位置 $\mathbf{x}$ における地形の高さ，a_{ext} は外力を表す．

本論文では，しぶきなどの 3 次元的な挙動を 3D パーティクルを用いて表現するため，Shallow Water シミュレーションもグリッド法でなく，パーティクル法を用いて解く．よって式 (1)，式 (2) を 2 次元の SPH 法を用いて解く [9]．SPH Shallow Water シミュレーションでは，グリッド毎に定義される高さ場の代わりに，図 1 のように各パーティクルに高さを定義し，以下の式を解くことで各パーティクルの高さを更新する．

$$h_i = \frac{\rho_i^{2D}}{\rho_0} \tag{3}$$

h_i，ρ_i はそれぞれパーティクル i の水深と密度，ρ_0 は水の初期密度である．式 (3) により式 (2) は以下のように書き換えられる．

$$\frac{\partial \mathbf{u}_{xz}^{2D}}{\partial t} = -\frac{g}{\rho_0}\nabla\rho_i^{2D} - g\nabla H(\mathbf{x}) + a_{ext} \tag{4}$$

$\mathbf{u}_{xz}^{2D}$ は 2D パーティクルの xz 方向の速度である．式 (4) により 2D パーティクルの速度 $\mathbf{u}_{xz}^{2D}$ と位置 $\mathbf{x}_{xz}^{2D}$ をタイムステップ幅 Δt ごとに更新する．また $x_y^{2D} = h_i + H(\mathbf{x}_{xz}^{2D})$ とする．SPH Shallow Water シミュレーションでは，パーティクルの位置が変化することにより，密度が変わり，密度の変化を式 (3) で高さに変換することで水面のシミュレーションが行える．

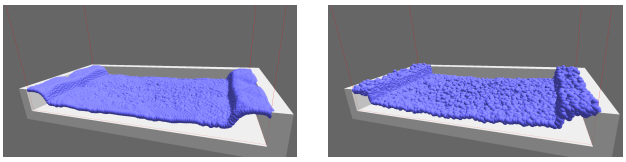
3.1.2 SPH Shallow Water の実装

安定化

2D パーティクルの安定したシミュレーションを行うために最大速度を $\sqrt{gh_i}$ でバインドする．$\sqrt{gh_i}$ は波動方程式から求めた，長波における水の伝播速度である．また安定化のために，Lee ら [8] と同様に Shallow Water モデルに粘性項 $\nu\nabla^2\mathbf{u}$ を追加する．$\nu(=\mu/\rho)$ は動粘性係数である．粘性項により式 (4) は以下の式 (5) のように書き換えられる．

$$\frac{\partial \mathbf{u}_{\mathbf{xz}}^{\mathbf{2D}}}{\partial t} = -\frac{g}{\rho^{3D}}\nabla\rho_i^{2D} - g\nabla H + \nu\nabla^2\mathbf{u}_{\mathbf{xz}}^{\mathbf{2D}} + a_{ext} \quad (5)$$

図 2 に粘性項の有無によるシミュレーション結果の違いを示す．



粘性項あり　　粘性項なし

図 2　粘性項の追加による安定化

境界処理

SPH 法ではシミュレーション境界や固体境界において，近傍パーティクルが少なくなり，境界付近の粒子の密度が小さくなってしまう．結果として密度が小さい粒子に他の粒子が集まり圧縮してしまうという問題点がある．そこでシミュレーション境界と固体内部に物理量の計算に用いる境界パーティクルを配置することで，境界付近で粒子の密度が小さくなることを防ぐ．またシミュレーション境界に流体パーティクルが接触した際には，速度を 0 としている．

近傍パーティクル探索

SPH 法では近傍の粒子によって物理量を求めるので近傍探索をする必要がある．総当たりで近傍探索を行うと，パーティクル数が N の時 $O(N^2)$ で計算コストがかかるので，近傍探索の高速化を行う．高速化にはハッシュを用いた空間分割法 [14] を CPU 上で実装して用いた．

3.2 3D パーティクルシミュレーション

3D パーティクルの運動には Position Based Fluids[5] を用いる．この手法はナビエ・ストークス方程式の圧力項を計算する代わりに，位置を繰り返し修正することで高速かつ高い非圧縮性を保つ位置ベース手法である．高い非圧縮性により，パーティクル間隔が保たれ，3D パーティクルと 2D パーティクルを合わせてレンダリングする際に違和感のないメッシュ形成が可能となる．

3.3 3D パーティクルの生成

Shallow Water モデルでは波の覆い被さりやしぶきなど

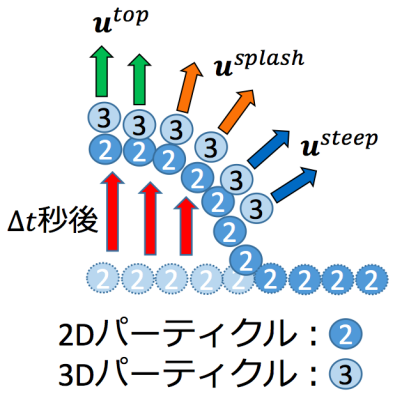


図 3　3D パーティクルの分類と初速度の方向

の 3 次元的な現象を表現することができない．このような現象を表現するために 3D パーティクルを生成する．これらの現象は，2D パーティクルの水深 h_i が大きく増加する，つまり水面が上昇する際に起こると考え，2D パーティクルが以下の条件式を満たすとき，3D パーティクルを生成する．

$$\frac{h_i - h_i^{prev}}{\Delta t} > \gamma_{3D} \quad (6)$$

h_i^{prev} は前ステップの 2D パーティクルの水深，γ_{3D} は 3D パーティクル生成のための閾値である．式 (6) の条件を満たしたパーティクルを近傍パーティクル j の最大高低差 $h_{diff} = \max(h_j) - \min(h_j)$ に応じて分類分けする．ここで $\max(h_j)$ は近傍 2D パーティクルの最大の高さ，$\min(h_j)$ は近傍 2D パーティクルの最小の高さである．

3D パーティクルを h_{diff} の値により Top，Splash，Steep パーティクルに分類する．図 3 に示すように，$h_{diff} \leqq \gamma_{top}$ の時 Top パーティクルを生成，$\gamma_{top} < h_{diff} < \gamma_{steep}$ の場合 Splash パーティクルを生成，$\gamma_{steep} \leqq h_{diff}$ である時 Steep パーティクルを生成する．ここで $0 \leqq \gamma_{top} \leqq \gamma_{steep}$ である．Top パーティクルは h_{diff} が小さい時，Steep パーティクルは h_{diff} が大きい時，Splash パーティクルは h_{diff} がこの二つの間の時に生成されるパーティクルである．

3D パーティクルの初期位置は分類によらず以下とする．

$$\mathbf{x}_{xz}^{3D} = \mathbf{x}_{xz}^{2D} \quad (7)$$

$$x_y^{3D} = (1 - \alpha_{height})h_i^{2D} + \alpha_{height}\max(h_j) \quad (8)$$

α_{height} は 3D パーティクルの生成高さを調整する係数である．本論文では，$\alpha_{height} = 0 \sim 0.3$ を用いている．それぞれのパーティクルの説明と初速度の計算方法を以下に示す．

- Top パーティクル
  図 3 に示すように，h_{diff} が小さい時，3D パーティクルは垂直に飛び出すとして速度を以下のように計算する．

$$\mathbf{u}_{xz}^{top} = 0 \quad (9)$$

$$u_y^{top} = k_{top}\frac{h_i - h_i^{prev}}{\Delta t} \quad (10)$$

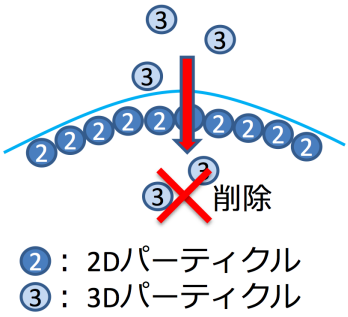


図 4　3D パーティクルの削除条件

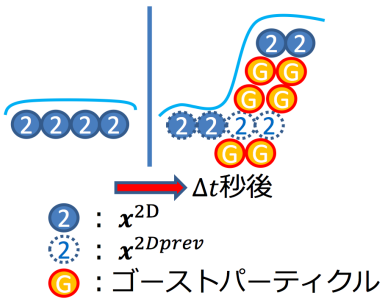


図 5　ゴーストパーティクルの配置例

ここで k_{top} は Top パーティクルの速度係数である．

- Steep パーティクル
  図 3 に示すように，h_{diff} が大きい時，3D パーティクルは斜めに飛び出すとして速度を以下のように計算する．

$$\mathbf{u}_{xz}^{steep} = \frac{a_{xz}^{2D}}{|a_{xz}^{2D}|}\sqrt{gh_i} \tag{11}$$

$$u_y^{steep} = k_{steep}\frac{h_i - h_i^{prev}}{\Delta t} \tag{12}$$

ここで k_{top} は Top パーティクルの速度係数，a_{xz}^{2D} は 2D パーティクルの xz 方向の加速度のベクトルである．式 (11) の右辺の $\sqrt{gh_i}$ は波動方程式から求めた，長波における水の伝播速度である．

- Splash パーティクル
  Splash パーティクルは Top パーティクルと Steep パーティクルの間にあり (図 3)，両パーティクルの速度を線形に補間することで，初速度を計算する

$$\mathbf{u}_{xyz}^{3D} = \alpha_{splash}\mathbf{u}_{xyz}^{steep} + (1-\alpha_{splash})\mathbf{u}_{xyz}^{top} \tag{13}$$

ここで α_{splash} は Splash パーティクルの補間係数であり，本論文では $\alpha_{splash} = (h_{diff} - \gamma_{top})/(\gamma_{steep} - \gamma_{top})$ とする．

3.4　3D パーティクルの削除

xz 平面上において 3D パーティクルの y 座標値が，近傍半径 l^{2D} 以内に存在する全近傍 2D パーティクルの y 座標値よりも低い場合，3D パーティクルの削除を行う (図 4)．削除の際に近傍半径 l^{2D} 以内に存在する近傍 2D パーティクルに対して，3D パーティクルの xz 方向の運動量と y 方向の運動量を保存するように 2D パーティクルの速度を以下の式 (14) で更新する．

$$\mathbf{u}_{xz}^{2D} = \frac{m_i^{2D}\mathbf{u}_{xz}^{2D} + \frac{m_i^{3D}}{N_{neigh}}\left(\mathbf{u}_{xz}^{3D} + |u_y^{3D}|\frac{\mathbf{r}_{xz}}{|\mathbf{r}_{xz}|}\right)}{m_i^{3D}} \tag{14}$$

N_{neigh} は近傍 2D パーティクルの個数，$\mathbf{r}_{xz}$ は xz 平面における 3D パーティクルから近傍 2D パーティクルへの相対位置を表している．$m_i^{2D}\mathbf{u}_{xz}^{2D}$ は元の 2D パーティクルの運動量であり，これに 3D パーティクルの運動量を加える．3D パーティクルの運動量 $m_i^{3D}u^{3D}$ を近傍 2D パーティクルに均等に分散させるために，質量 m_i^{3D} を近傍 2D パーティクルの個数 N_{neigh} で割っている．3D パーティクルの xz 方向の運動量 $(m_i^{3D}\mathbf{u}_{xz}^{3D}/N_{neigh})$ は，2D パーティクルの xz 方向の運動量としている．3D パーティクルの y 方向の運動量は 2D パーティクルを押しのける力として，2D パーティクルとの相対位置方向に変換している．

3.5　レンダリング

本論文では Screen Space Meshes(以下 SSM)[15] を用いてメッシュ生成を行う．この手法は 2D 空間でメッシュを生成するので，計算コストが低い．視点から見える部分のみにメッシュが生成されるので，ライティングの際に一回の屈折しか考慮できないという問題点があるが，提案手法ではシーン中の大部分において，表面のみにしかパーティクルがないためこの方法を用いた．

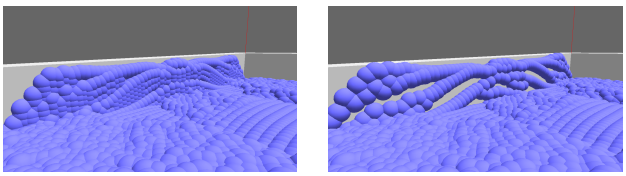
SSM はパーティクルがない部分にはメッシュが張られないため，2D パーティクルの動きの差が大きい場合に，2D パーティクルが疎になり，メッシュに穴があいてしまう．そこで，レンダリングの際に図 5 に示すようにゴーストパーティクルを生成しメッシュに穴があくことを防ぐ．ゴーストパーティクルを生成する条件は以下となる．

$$\gamma_{ghost} < |\mathbf{x}_{xyz}^{2D} - \mathbf{x}_{xyz}^{2Dprev}| \tag{15}$$

$\mathbf{x}_{xyz}^{2Dprev}$ は前ステップの 2D パーティクルの位置である．γ_{ghost} はゴーストパーティクル生成閾値である．ゴーストパーティクルは前ステップの 2D パーティクルの位置と現在の 2D パーティクルの位置を結ぶ直線上に配置する．配置する範囲は，$\mathbf{x}_{xyz}^{2Dprev} < \mathbf{x}_{xyz}^{2D}$ である時 $\mathbf{x}_{xyz}^{2D}$ から $\mathbf{x}_{xyz}^{2D} + k_{ghost}(\mathbf{x}_{xyz}^{2D^{prev}} - \mathbf{x}_{xyz}^{2D})$ まで，$\mathbf{x}_{xyz}^{2D} < \mathbf{x}_{xyz}^{2D^{prev}}$ である時 $\mathbf{x}_{xyz}^{2D^{prev}}$ から $\mathbf{x}_{xyz}^{2D^{prev}} + k_{ghost}(\mathbf{x}_{xyz}^{2D} - \mathbf{x}_{xyz}^{2D^{prev}})$ までとした．k_{ghost} はゴーストパーティクルの補間距離係数である．本論文では $k_{ghost} = 2$，$\gamma_{ghost} = r^{2D}/2$，ゴーストパーティクルを配置する間隔は r^{2D} としている．r^{2D} はパーティクル 2D の半径である．図 6 にゴーストパーティクルの有無によるシミュレーション結果の違いを示す．

表 1　シーンごとのパラメータ

	シーン 1	シーン 2	シーン 3	シーン 4	シーン 5
3D パーティクル高さ係数 α_{height}	0.3	0.3	0.3	0	0.3
3D パーティクル生成閾値 γ_{3D}	3	5	1.5	3	4
Top パーティクル生成閾値 γ_{top}	0.1	0.13	0.15	0.07	0.1
Steep パーティクル生成閾値 γ_{steep}	0.15	0.15	0.2	0.15	0.2
Top パーティクル速度係数 k_{top}	0.3	0.3	0.6	0.5	0.1
Steep パーティクル速度係数 k_{steep}	0.2	0.1	0.2	0.5	0



ゴーストパーティクルあり　　ゴーストパーティクルなし

図 6　ゴーストパーティクルによる補間

4. 実験結果と考察

本論文を CPU 上で実装し，図 7 に示す 5 つのシーンで実験を行った．実行環境は CPU:Intel Corei5-3470 3.20GHz, GPU:NVIDIA GeForce GTX 680 である．スクリーンサイズは $1,280 \times 720$，タイムステップ幅 $\Delta t = 0.005$ としている．3D パーティクル生成の際の閾値及び係数を表 1 に示す．

シーン 1 は中央の水が流れ出し，四方の壁で跳ね返り，中央で水柱があがる様子をシミュレーションしたものであり，図 7(a) に示すように激しいシーンにおいても 3 次元的な挙動が表現できていることがわかる．しかし，2D パーティクルが急激に集まった際に，密度が大幅に上昇し，現実と比べて大きな水柱が生じてしまう．これは Shallow Water モデルによるシミュレーションでは一定以上の激しい動きを正しく表現できないためであると考えられる．この問題点の対策としては，2D パーティクルに対して，高さ変化の上限を設定することが有効であると考えられる．図 7(b) のシーン 2 は中央にある水塊が左右の壁で跳ね返り，中央で再び水がぶつかる様子をシミュレーションしたものである．壁で水が跳ね返る際に水が覆いかぶさる現象，中央付近でぶつかる際にしぶきが垂直方向にあがる現象が観測できることから，3D パーティクルの生成の際の初期位置，初速度の場合分けが有効であると考えられる．図 7(c) に示すシーン 3 は複数の水滴が垂直に落下し，水面でしぶきをあげる様子をシミュレーションしたものである．図 7(d) のシーン 4 は，ななめに速度を持った水滴が落下し，しぶきをあげる様子をシミュレーションしたものある．これらのシーンではしぶきがあがる様子が確認できることから，垂直方向，水平方向の運動量保存が有効であると考えられる．図 7(e) のシーン 5 は，水が流れ出し，シミュレーション空間内にある物体でしぶきをあげる様子をシミュレーションしたものである．複雑な地形においても，3D パーティクルによりしぶきがあがることで，ある程度の現実的な振る舞いが観測できた．しかし複雑な地形においては，3D パーティクルの挙動が不自然なものとなる場合がある．これは，様々な水の流れが合流するような場合において，適切な初速度が設定できていないことが原因であると考えられる．

各シーンにおける計算時間を表 2 に示す．3D パーティクル数が多くなるほど計算コストが高くなっている事が分かる．

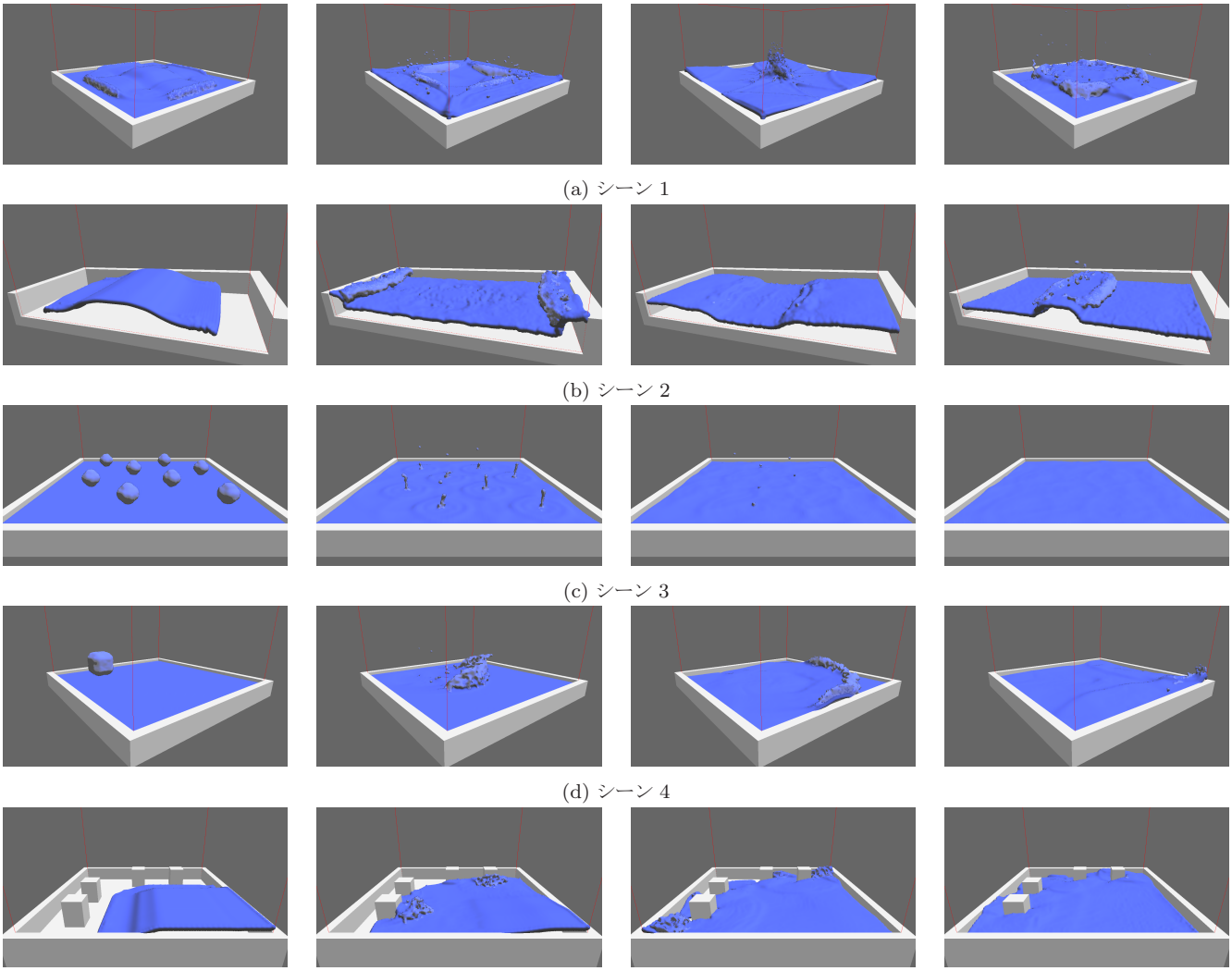
5. まとめと今後の課題

本論文では 2 次元のパーティクル法を用いた Shallow Water モデルによる波のシミュレーション結果に対して，3 次元のパーティクルを追加することで，3 次元的な挙動を表現できるシミュレーション手法を提案した．3 次元のパーティクルを発生時の状況により 3 種類に分類し，それぞれの運動を決定することで様々な表現を可能にした．そして 3 次元パーティクルの運動量を保存することで，現実的な振る舞いを可能にした．実験では，様々な状況において本論文が有効であることを示した．

今後の課題としてはメッシュ形成の問題があげられる．本論文では SSM[15] を用いているが，パーティクルが疎である箇所でゴーストパーティクルを用いてもまだ穴があいてしまうという問題がある．この問題を解決するための単純な方法は，ゴーストパーティクルを増やすことである．しかし，適切な位置にゴーストパーティクルを増やさなければ，本来と異なる水面が形成されてしまうので，生成位置をどのように求めるかを考える必要がある．またしぶきの生成条件に対する考察も不十分であり検討する必要がある．本論文では，閾値による生成，場合分けを行っているが，不自然な挙動のものがあった．この問題を解決するために，閾値を動的に変更するなどの対策が必要となる．また，提案手法では 3D パーティクル数が多い場合に計算速度が遅くなってしまう．そこで GPU を用いた並列計算により高速化を行い，リアルタイムでのシミュレーションを実現したい．

表 2　シーンごとの実行結果

	シーン 1	シーン 2	シーン 3	シーン 4	シーン 5
平均計算時間 (fps)	1.82	4.29	4.31	4.72	5.67
最大計算時間 (fps)	0.92	1.87	2.59	2.70	3.62
2D パーティクル数	13, 700	11, 200	10, 800	10, 800	24, 600
最大 3D パーティクル数	24, 774	12, 817	8, 000	8, 000	4, 264



(a) シーン 1

(b) シーン 2

(c) シーン 3

(d) シーン 4

(e) シーン 5

図 7　それぞれのシーンの実行結果

参考文献

[1] R. Bridson, Fluid Simulation for Computer Graphics, A K Petters, 2008.
[2] M. Müller, D. Charypar and M. Gross, “Particle-Based Fluid Fimulation for Interactive Applications”, In Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 154-159, 2003.
[3] M. Becker and M. Teschner, “Weakly Compressible SPH for Free Surface Flows” , In Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 209-217, 2007.
[4] B.Solenthaler and R. Pajorola, “Predictive-Corrective Incompressible SPH” , ACM Transaction on Graphics, Vol. 28, No. 3, pp. 40:1-40:6, 2009.
[5] M. Macklin and M. Müller, “Position Based Fluids” , ACM Transaction on Graphics, Vol. 32, No. 4, pp. 104:1-104:12, 2013.
[6] M. Kass and G. Miller, “Rapid, Stable Fluid Dynamics for Computer Graphics”, ACM SIGGRAPH Computer Graphics, Vol. 24, No. 4, pp. 49-57, 1990.
[7] A. T. Layton and M. Van De Panne, “A Numerically Efficient and Stable Algorithm for Animating Water Waves”, The Visual Computer, Vol. 18, No. 1, pp. 41-53, 2002.
[8] H. Lee and S. Han, “Solving the Shallow Water Equations Using 2D SPH Particles for Interactive Applications”, The Visual Computer, Vol. 26, No. 6-8, pp. 865-872, 2010.
[9] B. Solenthaler, P. Bucher, N. Chentanez, M. Müller and M.Gross, “SPH Based Shallow Water Simulation”, In Proceedings of Virtual Reality Interactions and Physical Simulations, pp. 39-46, 2011.
[10] N. Chentanez, M. Müller, S. Schirm and M. Gross,

"Real-Time Simulation of Large Bodies of Water with Small Scale Details", In Proceedings of ACM SIGGRAPH / EUROGRAPHICS Symposium on Computer Animation, pp. 197-206, 2010.

[11] N. Thuerey, M. Müller, S. Schirm and M. Gross, "Real-time Breaking Waves for Shallow Water Simulations", In Proceedings of the 15th Pacific Conference on Computer Graphics and Applications, pp. 39-46, 2007.

[12] N. Chentanez, M. Müller and T.Y. Kim, "Coupling 3D Eulerian, Height Field and Particle Methods for the Simulation of Large Scale Liquid Phenomena", In Proceedings of ACM SIGGRAPH / EUROGRAPHICS Symposium on Computer Animation, pp. 1-10, 2014.

[13] H. Wang, G. Miller and G. Turk, "Solving General Shallow Wave Equations on Surfaces", In Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 229-238, 2007.

[14] S. Green, "Cuda particles", nVidia Whitepaper, 2008.

[15] M. Müller, S. Schirm and S. Duthaler, "Screen Space Meshes", In Proceedings of ACM SIGGRAPH / EUROGRAPHICS Symposium on Computer Animation, 2007.